อาจหาญ สัตยารักษ์

SOFT MAIL

สร้างโปรแกรม เรสซิเดนต์ : พิมพ์ภาพกราฟิก

เขียนโปรแกรมเรสซิเดนต์ แบบง่าย ๆ ไว้ใช้เอง ด้วยเทอร์โบปาสคาล

หลังจากที่ผู้เขียนได้เสนอโปรแกรมพิมพ์ภาพกราฟิก โดยลอกจากหน้าจอภาพในบทความชื่อ พิมพ์ภาพกราฟิกด้วยปาสคาล 4.0 ในฉบับที่ 44 หน้า 195-198 นั้น ก็ช่วยให้การพิมพ์ภาพในโปรแกรมของผู้พัฒนาบนเทอร์โบปาสคาลเป็นเรื่องที่ง่ายขึ้น โดยการกำหนดชุดคำสั่งการพิมพ์ไว้ในโปรแกรมนั้น ๆ เลย ต่อมาเมื่อเครื่องพิมพ์แบบ 24 หัวเข็มนิยมใช้กันมากขึ้น ก็มีผู้พัฒนาชุดคำสั่งพิมพ์ภาพด้วยเครื่องพิมพ์ 24 หัวเข็มเพิ่มเติม เช่นในวารสารไมโครคอมพิวเตอร์ฉบับที่ 84 เรื่อง แสดงภาพกราฟิกความเร็วสูงด้วยปาสคาล โดย คุณเรืองเดช สุทธิพล ในหน้า 293 เป็นต้น

แต่ปัญหาก็เกิดขึ้นจนได้ เพราะมีผู้อ่านหลายท่านอยากพิมพ์ภาพกราฟิกบนจอภาพ แต่ในโปรแกรมนั้น ๆ ไม่มีการกำหนดพิมพ์ภาพในตนเอง ซึ่งความจริงแล้วเราสามารถใช้ไฟล์ของดอสที่ชื่อ graphics.com ช่วยได้โดยป้อนพารามิเตอร์ให้แก่ไฟล์นี้ ตัวอย่าง (ดอส 5.0) เช่น

```
A>graphics color1 พิมพ์ด้วยเครื่องพิมพ์สีเดียว
A>graphics color4 พิมพ์ด้วยเครื่องพิมพ์สามสี (RGB)
A>graphics color8 พิมพ์ด้วยเครื่องพิมพ์แปดสี
A>graphics graphics พิมพ์ด้วยเครื่องพิมพ์ทั่วไป
ฯลฯ
```

โปรแกรมนี้เป็นการพิมพ์โดยใช้ข้อมูลที่มีในหน่วยความจำส่วนควบคุมการแสดงบนจอภาพ (ขอให้อ่านรายละเอียดการใช้งานเพิ่มเติมจากหนังสือคำอธิบายเรื่องดอสเล่มอื่น ๆ)

การทำงานของไฟล์ graphics.com นี้จะเป็นโปรแกรมแบบฝังตัวในหน่วยความจำหรือที่เรียกเป็นโปรแกรมเรสซิเดนต์

วันนี้...เราจะมาสร้างโปรแกรมเรสซิเดนต์กันบ้าง โดยผู้เขียนขอเสนอเป็น 2 ตอน คือ การพิมพ์ภาพและการเก็บภาพหน้าจอ โดยจะพูดเรื่องการพิมพ์ภาพก่อน

อินเตอร์รัพต์ของแป้นพิมพ์

สำหรับการสร้างโปรแกรมเรสซิเดนต์ที่ทำงานด้วยการตรวจการกดแป้นพิมพ์นี้ เราต้องอาศัยการทำงานอินเตอร์รัพต์ (interrupt) หมายเลข $9 ซึ่งเป็นอินเตอร์รัพต์ทางฮาร์ดแวร์ที่ตรวจการกดแป้นพิมพ์เพื่อส่งข้อมูลไปให้แก่ซีพียู (CPU) รับรู้ โดยอาศัยหลักการสำคัญดังนี้

1. เก็บรายละเอียดเดิมของอินเตอร์รัพต์นี้ไว้เพื่อนำมาคืนในภายหลัง

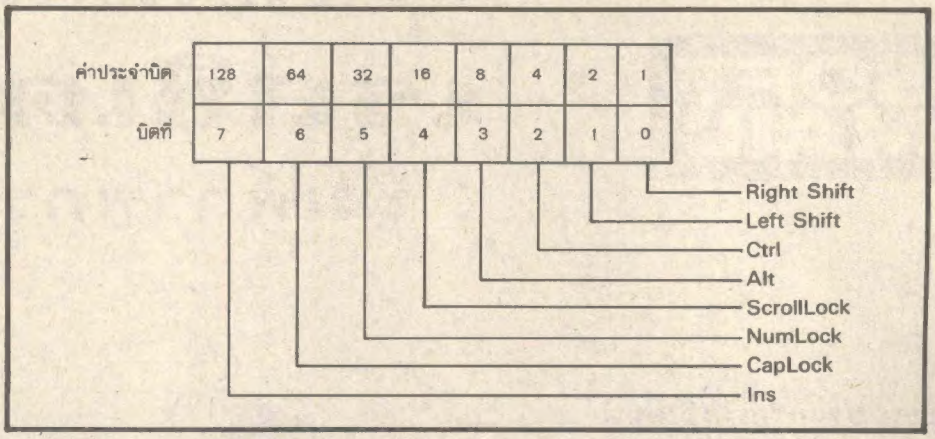
2. กำหนดการทำงานของอินเตอร์รัพต์นี้ใหม่เพื่อที่เราจะสามารถเขียนโปรแกรมตรวจการกดแป้นพิมพ์ให้ทำงานตามต้องการของเราได้

3. เมื่อยกเลิกการทำงานก็ให้คืนรายละเอียดเดิม (ข้อ 1) คืนให้แก่หน่วยความจำเพื่อที่ให้อินเตอร์รัพต์นี้กลับไป

ทำงานตามปกติต่อไป โดยอาศัยคำสั่ง getIntVec, setIntVec มาช่วยงาน

การตรวจสถานะการกดแป้นพิมพ์พิเศษ

สำหรับการตรวจสอบการกดแป้นพิมพ์พิเศษนั้น เราอาศัยข้อมูลของแอดเดรส $0040:$0017 ที่ควบคุมสถานะการกดแป้นเหล่านี้โดยแต่ละบิตจะใช้ในการควบคุมแต่ละแป้นจึงสามารถควบคุมได้บิตละหนึ่งแป้นพิมพ์ โดยตรวจสอบสถานะการกดแป้นได้จากรูปที่ 1 ดังนี้



รูปที่ 1 แสดงข้อมูลแต่ละบิตของแอดเดรส $0040:$0017 ที่ควบคุมสถานะการกดแป้นพิเศษ

สมมุติให้ตัวแปร KbdFlag รองรับค่าที่อ่านได้จากแอดเดรสนี้ ก็จะมีความหมายดังนี้

ถ้า KbdFlag= 0 แสดงว่าทั้งแปดแป้นพิมพ์นั้นอยู่ในสถานะปกติ

ถ้า KbdFlag= 1 แสดงว่าแป้นพิมพ์ Right Shift ถูกกด

ถ้า KbdFlag= 2 แสดงว่าแป้นพิมพ์ Left Shift ถูกกด

ถ้า KbdFlag= 4 แสดงว่าแป้นพิมพ์ Ctrl ถูกกด

ถ้า KbdFlag= 8 แสดงว่าแป้นพิมพ์ Alt ถูกกด

ถ้า KbdFlag= 16 แสดงว่าแป้นพิมพ์ ScrollLock อยู่ในสถานะทำงาน

ถ้า KbdFlag= 32 แสดงว่าแป้นพิมพ์ NumLock อยู่ในสถานะทำงาน

ถ้า KbdFlag= 64 แสดงว่าแป้นพิมพ์ CapLock อยู่ในสถานะทำงาน

ถ้า KbdFlag= 128 แสดงว่าแป้นพิมพ์ Ins อยู่ในสถานะทำงาน

ถ้ากดแป้น Alt, Left Shift พร้อมกันก็จะได้ค่า KbdFlag= 8+2 = 10

ถ้ากดแป้น Alt, Right Shift พร้อมกันก็จะได้ค่า KbdFlag= 8+1 = 9

เราสามารถตรวจการกดแป้นกลุ่มนี้ได้ตามความต้องการ เช่นในโปรแกรมที่จะกล่าวต่อไปนี้ได้กำหนดการทำงานดังนี้

กดแป้น Alt, Left Shift พร้อมกันจะพิมพ์กราฟิกหน้า 0

กดแป้น Alt, Right Shift พร้อมกันจะพิมพ์กราฟิกหน้า 1

กดแป้น Ctrl, left Shift หรือ Ctrl, Right Shift พร้อมกันจะยกเลิกการทำงาน

รวมไฟล์จัดการจอภาพเข้ากับโปรแกรม

เพื่อให้ไฟล์ *.exe ที่ได้มาใช้งานนั้นมีเพียงไฟล์เดียว เราจึงมีความจำเป็นต้องนำไฟล์ *.bgi มารวมไว้กับไฟล์โปรแกรมที่เขียนนั้น ๆ ด้วย ซึ่งสามารถทำได้ดังนี้

1. เปลี่ยนไฟล์จัดการจอภาพที่มีสกุลเป็น bgi ให้เป็นไฟล์สกุล obj ก่อน โดยอาศัยไฟล์ที่ให้มาพร้อมแผ่นต้นฉบับของเทอร์โบปาสคาลที่ชื่อ BINOBJ.EXE ซึ่งมีรูปแบบการเรียกใช้ดังนี้

```
binobj ชื่อไฟล์ *.bgi ชื่อไฟล์เมื่อเป็น *.obj ชื่อชุดคำสั่งเรียกในปาสคาล
```

สมมติเราต้องการนำไฟล์ herd.bgi และ egavga.bgi มารวมไว้ในไฟล์ *.EXE ของเรา ก็จะทำตามขั้นตอนดังนี้

1.1 ใส่แผ่นที่มีไฟล์ binobj.exe และไฟล์ *.bgi ในช่องอ่านแผ่นบันทึก A:

1.2 พิมพ์ผ่านดอสดังนี้

```
A>binobj herc.bgi hercdrv hercdriver
A>binobj egavga.bgi vgadrv egavgadriver
```

1.3 เมื่อจบการทำงานทั้งสองครั้งจะได้ไฟล์ hercdrv.obj, vgadrv.obj ในแผ่นบันทึก ทั้งสองไฟล์นี้ยังไม่สามารถเรียกใช้ในปาสคาลได้โดยตรง

2. เขียนไฟล์บนปาสคาลดังชุดคำสั่งในรูปที่ 2 บรรทัดที่ 12-15

3. การเข้าสู่กราฟิกจะต้องเขียนชุดคำสั่งขึ้นมาโดยเฉพาะ ดังชุดคำสั่ง

```
 1: PROGRAM Print_Graphic_Screen;
 2:  { Rasident Program
 3:    Alt+left_Shift      Print Graphic Page 0
 4:    Alt+Right_Shift     Print Graphic Page 1
 5:    Ctrl+Shift          Quit Rasident Program }

 6: {$M 2400,0,8000}
 7: uses dos,crt,graph;
 8: var kbdFlag:byte Absolute $0040:$0017;
 9:     Int9Save:Pointer;
10:     Regs:registers;   ch:char;
11:     VGA480:boolean;  Drive:string;

12: PROCEDURE HercDriver; external;
13:   {$L Hercdrv.obj}
14: PROCEDURE EgaVgaDriver; external;
15:   {$L Vgadrv.obj}

16: PROCEDURE Opengraph;
17: var grdriver,Graphck,Grmode:integer;
18:     Mono:boolean;
19: begin
20:    Mono:=false;
21:    detectgraph(grdriver,grmode);
22:    case grdriver of
23:      7:Mono:=true;
24:      9:If Pred(VGA480) then grmode:=1 Else grmode:=2;
25:    end;
26:  if Mono then Graphck:=RegisterBgidriver(@HercDriver)
27:        else   Graphck:=RegisterBgidriver(@EgavgaDriver);
28:   initgraph(grdriver,grmode,'');
29: end;

30: PROCEDURE CopyByte(ByteSt :string);
31: var  B:byte;
32: begin
33:     for b:=1 to length(ByteSt) do
34:      begin
35:       Regs.AH := 0;        Regs.AL := ord(ByteSt[b]);;
36:       Regs.DX := Pred(1);  Intr($17, Regs);
37:      end;
38: end;
```

รูปที่ 2 การกำหนดตัวแปรและชุดคำสั่งส่งข้อมูลไปพิมพ์ลงกระดาษ

opengraph ในรูปที่ 2

ในซอฟต์เมล์ได้ให้ไฟล์ hercdrv.obj, vgadrv.obj มาด้วยแล้ว รายละเอียดของรูปที่ 2 มีดังนี้

1. บรรทัดที่ 1-5 บอกรายละเอียดของแป้นที่จะใช้งาน

2. บรรทัดที่ 6 กำหนดหน่วยความจำที่จะใช้งานเพื่อให้เป็นเรสซิเดนต์โปรแกรม

3. บรรทัดที่ 7-11 กำหนดยูนิตที่ใช้งานร่วมและตัวแปรที่ต้องใช้

4. บรรทัดที่ 12-15 ชุดคำสั่งนำไฟล์จัดการจอภาพสำหรับจอโมโนโครมและจอสีหรือจอวีจีเอ มารวมไว้ในโปรแกรม *.exe

5. บรรทัดที่ 16-29 การเข้ากราฟิกโหมดโดยไม่อาศัยไฟล์ *.bgi

6. บรรทัดที่ 30-38 ชุดคำสั่งส่งข้อมูลไปพิมพ์ลงกระดาษ

การพิมพ์ภาพกราฟิก

เพื่อให้เห็นแนวทางการเขียนโปรแกรมอีกแบบหนึ่ง ผู้เขียนจึงขอเสนอแนวคิดในการลอกจอภาพด้วยเครื่องพิมพ์แบบ 24 หัวเข็ม ซึ่งตามแนวคิดนี้จะสามารถลอกภาพหน้าจอกราฟิกขนาดใหญ่สุดคือ 640×480 จุด ลงกระดาษได้ภาพในขนาดที่ค่อนข้างเป็นปกติ

ก่อนอื่นขอให้ท่านย้อนไปอ่านแนวการพิมพ์ของเครื่องพิมพ์แบบ 24 หัวเข็มจากบทความ เรื่อง การพิมพ์อักขระแบบกราฟิก ตอนที่ 2 จากวารสารไมโครคอมพิวเตอร์ฉบับที่ 64 หน้า 286 เพื่อประกอบความเข้าใจในเรื่องการพิมพ์แบบกราฟิก แนวคิดมีดังนี้

1. อ่านข้อมูลมาแบบเดียวกันกับการพิมพ์ด้วยเครื่องพิมพ์ 9 หัวเข็ม จะได้ข้อมูลชนิดไบต์

2. นำข้อมูลที่ได้มาแบ่งเป็น 2 ส่วนคือ สี่บิตสูงและสี่บิตต่ำ

3. นำข้อมูลสี่บิตสูงมาขยายให้เป็นข้อมูลหนึ่งไบต์

4. นำข้อมูลสี่บิตต่ำมาขยายให้เป็นข้อมูลหนึ่งไบต์

5. นำข้อมูลในข้อ 3 และ 4 พร้อมกันข้อมูลค่า 0 รวมเป็นข้อมูล 3 ไบต์ ส่งไปให้เครื่องพิมพ์

6. ทำแบบนี้กับข้อมูลอื่น ๆ จนสิ้นสุดภาพที่จะพิมพ์

การพิมพ์ด้วยคำสั่งนี้จะได้ภาพมีความสูงประมาณ 16/180 นิ้วต่อการพิมพ์หนึ่งแถว (8 จุดในแนวตั้ง) ทำให้พิมพ์ภาพกราฟิกสูง 480 จุดจะได้ภาพมี

ความสูงเพียง (480/8)×(16/180) = 5.33 นิ้ว ดังนั้นชุดคำสั่งนี้จึงใช้ในการลอกหน้าจอของกราฟิกขนาดใหญ่ได้พอดี (ตัวอย่างคือรูปที่ 3) หากพิมพ์ภาพขนาด 640×348 ก็จะได้ภาพมีความสูง (348/8)×(16/180) = 3.87 นิ้ว (รูปที่ 4)

อาจใช้ชุดคำสั่งจากบทความเรื่อง พิมพ์ภาพกราฟิกด้วยปาสคาล 4.0 ฉบับที่ 44 หน้า 195-198 หรือจากฉบับที่ 84 เรื่อง แสดงภาพกราฟิกความเร็วสูงด้วยปาสคาล ของคุณเรืองเดช สุทธิพล ในหน้า 293 มาใช้งานแทนได้ (ในซอฟต์เมล์มีไฟล์ hardcopy.pas สำหรับเครื่องพิมพ์ 9 หัวเข็มด้วย)

นำแนวคิดข้างต้นมาเขียนชุดคำสั่งได้ดังรูปที่ 5 มีรายละเอียดดังนี้

1. บรรทัดที่ 41-50 เป็นฟังก์ชันในการขยายข้อมูลสี่บิตสูงให้เป็นหนึ่งไบต์ โดยทำให้ บิตที่ 7-6, บิต 5-4, บิต 3-2 และ บิต 1-0 เหมือนกัน โดยขอให้ดูค่าคงที่ที่กำหนดไว้ จะคู่กันดังนี้

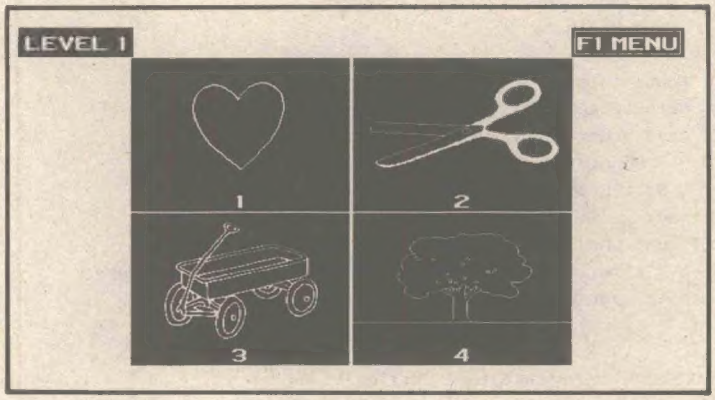
ข้อมูลเดิม		ข้อมูลใหม่
บิตที่ 7	128	192
บิตที่ 6	64	48
บิตที่ 5	32	12
บิตที่ 4	16	3

2. บรรทัดที่ 51-60 คือฟังก์ชันในการขยายข้อมูลสี่บิตต่ำให้เป็นหนึ่งไบต์ จะมีลักษณะดังนี้

ข้อมูลเดิม		ข้อมูลใหม่
บิตที่ 3	8	192
บิตที่ 2	4	48
บิตที่ 1	2	12
บิตที่ 0	1	3

3. บรรทัดที่ 61-87 คือชุดคำสั่งในการนำข้อมูลที่อ่านได้ไปลงพิมพ์ มีรายละเอียดดังนี้

รูปที่ 3
ภาพที่ลอกจากจอภาพ ขนาด 640×480 จุดด้วยเครื่องพิมพ์ 24 เข็ม



รูปที่ 4
ภาพที่ลอกจากจอภาพ ขนาด 640×348 จุดด้วยเครื่องพิมพ์ 24 เข็ม

3.1 บรรทัดที่ 68-74 ส่วนการตรวจสอบหาจุดสุดท้ายที่จะส่งข้อมูลไปพิมพ์ โดยเป็นการตรวจจากด้านขวามือมายังซ้ายมือ ทั้งนี้จะช่วยการพิมพ์มีความเร็วมากขึ้น

3.2 บรรทัดที่ 76-77 คำนวณหาจำนวนจุดที่จะส่งไปพิมพ์ พร้อมกับเข้ารหัสการพิมพ์แบบกราฟิกซึ่งจะต้องกำหนดให้ทราบด้วยว่าจะส่งข้อมูลไปจำนวนเท่าใด ข้อมูลที่ส่งไปจริง ๆ จะมีขนาดเป็น 3 เท่าของขนาดที่คำนวณได้ ทั้งนี้เพราะข้อมูลหนึ่งไบต์จะควบคุม 8 หัวเข็ม โปรดสังเกตว่าผู้เขียนได้กำหนดระยะบรรทัดไว้แค่ 16/180 นิ้วเท่านั้น (Copybyte(#27'3'#16)

3.3 บรรทัดที่ 78-86 การส่งข้อมูลที่อ่านได้ไปพิมพ์ทั้งนี้จะต้องเปลี่ยนข้อมูลหนึ่งไบต์ให้เป็นข้อมูล 2 ไบต์เสียก่อน (กำหนดให้ไบต์ที่สามเป็นศูนย์) โดยทำการตรวจข้อมูลครั้งละ 8 จุดตามแนวตั้งจึงได้ข้อมูลหนึ่งไบต์ นำไปสั่งพิมพ์ลงกระดาษด้วยการเรียกใช้ฟังก์ชันการขยายสี่บิตสูงและสี่บิตต่ำ และจบลงด้วยการสั่งให้เลื่อนกระดาษขึ้นหนึ่งบรรทัด คือคำสั่ง (copybyte (#10)

4. บรรทัดที่ 89-90 กำหนดขนาดของจอภาพที่จะพิมพ์และตั้งกั้นซ้าย (ตั้งไว้ที่ #10 แต่เปลี่ยนเป็นค่าอื่นได้ตามความต้องการ)

```
39: PROCEDURE HardCopy;   {สำหรับเครื่องพิมพ์แบบ 24 หัวเข็ม }
40: var Endscan,Scanline,n1,n2:word;
41:     FUNCTION UP4 TOByte(B:byte):Byte;
42:      const BB:array[4..7] of byte=(3,12,48,192);
43:            bck:array[4..7] of byte=(16,32,64,128);
44:      var k,Bval:byte;
45:       begin
46:         BVal:=0;
47:         for k:=4 to 7 do
48:         if B and Bck[k]=Bck[k] then Bval:=Bval+BB[k];
49:         UP4 TOByte:=Bval;
50:       end;
51:     FUNCTION DOWN4 TOByte(B:byte):Byte;
52:      const BB:array[0..3] of byte=(3,12,48,192);
53:            bck:array[0..3] of byte=(1,2,4,8);
54:      var k,Bval:byte;
55:       begin
56:         BVal:=0;
57:         for k:=0 to 3 do
58:         if B and Bck[k]=Bck[k] then Bval:=Bval+BB[k];
59:         DOWN4 TOByte:=Bval;
60:       end;
61:     PROCEDURE CopyDot:
62:       const
63:         Bits : array[0..7] of byte = (128,64,32,16,8,4,2,1);
64:       var
65:         Bit,PrintByte: byte; MaxX:word;
66:         XYPixel: integer;
67:     begin
68:          MaxX:=GetMaxX+1;
69:          Repeat
70:              Dec(MaxX); PrintByte:=0;
71:              for Bit := 0 to 7 do
72:                   if (GetPixel(MaxX,Scanline*8+bit) <> 0) then
73:                              PrintByte := PrintByte+ Bits[Bit];
74:         until (PrintByte<>0) or (MaxX=0);
75:        if MaxX<>0 then      begin
76:         n1 := Lo(Succ(MaxX));  n2 := Hi(Succ(MaxX));
77:         copyByte(#27'3'#16#27'*'#38+chr(n1)+chr(n2));
78:           for XyPixel:=0 to MaxX do
79:             begin  PrintByte := 0;
80:              for Bit := 0 to 7 do
81:                if  (GetPixel(XyPixel,Scanline*8+bit) <> 0) then
82:                           PrintByte := PrintByte+ Bits[Bit];
83:              CopyByte(chr(UP4_TOByte(PrintByte)));
84:              CopyByte(chr(DOWN4_TOByte(PrintByte))+#0);
85:             end;          end;
86:              CopyByte(#10);
87:     end;

88: begin
89:    Endscan:=GetMaxY;  CopyByte(#27'1'#10);
90:    if VGA480 then EndScan:=480;
91:    ScanLine:=0;
92:     Repeat
93:       CopyDot;  Inc(Scanline);
94:     until ScanLine>(EndScan div 8);
95:   Copybyte(#27#2#27'1'#0#27'<');
96: end;
```

รูปที่ 5 ชุดคำสั่งในการออกจอภาพลงกระดาษ

5. บรรทัดที่ 91-96 ส่วนการส่งข้อมูลจากบนสุดของจอภาพไปพิมพ์และสิ้นสุดการพิมพ์ที่ด้านล่างสุดของจอภาพ

การกำหนดแป้นพิมพ์ใช้งาน

เพื่อให้การทำงานเป็นไปโดยอัตโนมัติเมื่อตรวจพบการกดแป้นพิมพ์ที่ต้องการ เราจำเป็นต้องเขียนส่วนการตรวจการกดแป้นพิมพ์นี้ ให้ตรงกับความต้องการซึ่งจะมีดังนี้

ถ้ากดแป้น Alt และ Left Shift พร้อมกันจะเป็นการพิมพ์กราฟิกหน้า 0

ถ้ากดแป้น Alt และ Right Shift พร้อมกันจะเป็นการพิมพ์กราฟิกหน้า 1

ถ้ากดแป้น Ctrl และ Shift (ซ้ายหรือขวาก็ได้) พร้อมกันจะหยุดการทำงานของโปรแกรม

เพื่อให้การตรวจการกดแป้นพิมพ์เป็นแบบโปรแกรมเรสซิเดนต์ ต้องอาศัยการเขียนคำสั่งในรูปแบบของภาษาแอสเซมบลีมาช่วยงาน เนื่องจากผู้เขียนไม่มีความรู้นี้โดยตรงจึงได้อาศัยชุดคำสั่ง Int9Handler จากหนังสือ เทคนิคการเขียนโปรแกรมชั้นสูง "แอดวานซ์เทอร์โบปาสคาล version 4.0" ซึ่งเขียนโดย อ.สุรศักดิ์ สงวนพงษ์ พิมพ์และจัดจำหน่ายโดยบริษัท ซีเอ็ดยูเคชั่น จำกัด รายละเอียดอยู่ในหน้า 143-145 มาเป็นแม่แบบผู้เขียนจึงเรียนขออนุญาตทั้งผู้เขียนและผู้พิมพ์-จัดจำหน่ายมาพร้อมนี้ด้วย สำหรับหนังสือนี้ผู้อ่านจะหาความรู้ในเรื่องอื่น ๆ ได้อีกหลายประการ ดังนั้นหากยังไม่มีหนังสือเล่มนี้ก็ควรซื้อมาศึกษาเพิ่มเติมความรู้ของตนเองก็จะเกิดประโยชน์อย่างมาก

เมื่อนำมาเขียนโปรแกรมจะได้ชุด

คำสั่งดังรูปที่ 6

ผู้เขียนขออนุญาตอธิบายรายละเอียดเฉพาะชุดคำสั่ง UpDateStatus ดังนี้

1. บรรทัดที่ 99-103 ตรวจสอบการกดแป้น Alt+Left shift เพื่อพิมพ์ภาพกราฟิกที่อยู่หน้า 0

2. บรรทัดที่ 104-108 ตรวจสอบการกดแป้น Alt-Right shift เพื่อพิมพ์ภาพกราฟิกที่อยู่หน้า 1

3. บรรทัดที่ 109-114 ตรวจสอบการกดแป้น Ctrl+shift (ซ้ายหรือขวาก็ได้) เพื่อเลิกงาน

หากผู้อ่านต้องการให้พิมพ์แบบเลือกชนิดเครื่องพิมพ์ได้ทั้งแบบ 9 และ 24 หัวเข็ม ทำดังนี้

1. เพิ่มชุดคำสั่งการพิมพ์แบบ 9 เข็มเข้ามาในโปรแกรม สมมติให้ชื่อชุดคำสั่งว่า hardcopy09

2. เพิ่มตัวแปรบูลีนชื่อ Print24 เพื่อใช้ตรวจสอบการใช้เครื่องพิมพ์ (กำหนดเป็นตัวแปรโกบอล)

3. เพิ่มพารามิเตอร์เพื่อกำหนดให้เป็นการพิมพ์แบบ 9 หัวเข็มหรือ 24 หัวเข็มในตอนเรียกใช้ไฟล์ซึ่งจะทำให้ตัวแปร Print24 เป็นจริงเมื่อต้องการใช้เครื่องพิมพ์แบบ 24 หัวเข็ม (เพิ่มเติมที่โปรแกรมหลัก)

4. ต้องเปลี่ยนคำสั่งในบรรทัดที่ 102 และ 107 มาเป็นแบบกำหนดทางเลือกคือเปลี่ยนทั้งบรรทัดเป็น if Print24 then hardcopy else hardcopy09

```
 97: PROCEDURE UpDateStatus;
 98: begin
 99:   if (KbdFlag and 10)=10 then
100:    begin
101:      setactivepage(0);
102:      HardCopy;
103:    end;
104:   if (KbdFlag and 9)=9 then
105:    begin
106:      setactivepage(1);
107:      HardCopy;
108:    end;
109:   if ((KbdFlag and 6)=6) or ((KbdFlag and 5)=5) then
110:    begin
111:      setIntVec(9,int9Save);
112:      closeGraph;
113:      halt;
114:    end;
115: end;

116: {$F+}
117: PROCEDURE Int9Handler;
118: {$F-}
119: Interrupt;
120: begin
121:   Inline($9C/
122:          $3E/$FF/$1E/Int9Save/
123:          $FA);
124:    UpDateStatus;
125:   Inline($FB);
126: end;
```

รูปที่ 6 ชุดคำสั่งตรวจสอบการกดแป้นเพื่อนำไปกำหนดการพิมพ์ภาพกราฟิกหน้าจอ

ส่วนของโปรแกรมหลัก

เพื่อให้ผู้เรียกใช้ไฟล์นี้ทราบถึงความสามารถในการทำงานและเงื่อนไขของการใช้แป้นพิมพ์ เราจึงต้องเขียนรายละเอียดให้ผู้ใช้ทราบเมื่อรันโปรแกรมพร้อมกำหนดทางเลือกให้โดยการป้อนพารามิเตอร์ให้ตรงกับความต้องการ ซึ่งจะเขียนโปรแกรมได้ดังรูปที่ 7

มีรายละเอียดดังนี้

1. บรรทัดที่ 129 เก็บชื่อพาธที่รันไฟล์นี้ไว้ในตัวแปร Drive เพื่อนำไปใช้ในคำอธิบายการใช้งาน

2. บรรทัดที่ 130-144 แสดงคำอธิบายรายละเอียดการทำงานและเงื่อนไขการใช้แป้นต่าง ๆ ตลอดจนแจ้งให้ทราบถึงพารามิเตอร์ที่ใช้ประกอบ หากต้องการกำหนดชนิดเครื่องพิมพ์ด้วยให้กำหนดที่พารามิเตอร์ตัวที่สอง

3. บรรทัดที่ 145-153 หากกดแป้น S หรือ s ก็จะเป็นการฝังโปรแกรมนี้ลงในหน่วยความจำให้เป็นโปรแกรมเรสซิเดนต์ต่อไป โดยจะทำการเข้าสู่โหมดกราฟิกตามที่กำหนดไว้ พร้อมกับจะวนรอดักการกดแป้นตามที่เรากำหนดไว้ให้ทำงาน

โปรแกรมสำหรับบทความนี้ในซอฟต์เมล์ใช้ชื่อว่า printscn.pas ซึ่งเมื่อผู้อ่านคอมพายล์ก็จะได้ไฟล์ printscn.exe มาใช้งาน

เมื่อรันโปรแกรมนี้ในช่วงแรกหลังการกดแป้น S หรือ s นั้นโปรแกรมจะอยู่ในกราฟิกโหมด จึงจะไม่แสดงเคอร์เซอร์ในปรากฏ แต่ท่านสามารถป้อนชื่อไฟล์กราฟิกที่ต้องการลอกภาพหน้าจอได้ทันที

```
127: {========================== MAIN ====================}
128: begin
129:   getdir(0,drive);    Drive:=Drive+'>';
130:   clrscr;
131:   writeln('Rasident Program for Print Graphics Screen');
132:   writeln('=== By ARTHAN SATTAYARAK Lampang 52000 ===');
133:   writeln;
134:   writeln('Alt+left_Shift      Print Graphic Page 0');
135:   writeln('Alt+Right_Shift     Print Graphic Page 1');
136:   writeln('Ctrl+Shift          Quit Rasident Program');
137:   writeln;
138:   writeln('      === Graphics 640x350 or 720x348 ===');
139:   writeln('usage : '+Drive+'Printgr');
140:   writeln('      === Graphics 640x480 (VGA only) ===');
141:   writeln('usage : '+Drive+'Printgr 480');
142:   writeln;
143:   writeln('press  S  == for Install Program');
144:   ch:=readkey;
145:  if Upcase(ch)='S' then
146:   begin
147:     Opengraph;
148:     VGA480:=false; if Paramstr(1)='480' then VGA480:=true;
149:     UpDateStatus;
150:     getIntVec(9,Int9Save);
151:     setIntVec(9,@int9handler);
152:     keep(0);
153:  end;
154: end.
```

รูปที่ 7 ส่วนของโปรแกรมหลักแสดงรายละเอียดของการใช้งาน

เนื่องจากโปรแกรมนี้เขียนด้วยภาษาชั้นสูง ดังนั้นโปรแกรมจะใช้หน่วยความจำมากนิดหนึ่ง จึงอาจมีปัญหาในการรันไฟล์กราฟิกบางไฟล์ แต่เมื่อผู้เขียนได้ทดสอบกับไฟล์ที่เขียนด้วยภาษาปาสคาลและเกมบางเกมก็พบว่าสามารถรันโปรแกรมและลอกจอภาพได้

การใช้งานก็ทำดังนี้

1. รันไฟล์ printscn.exe ดังนี้ (สมมติอยู่ในไดรฟ์ A)

```
A>printscn    ถ้าต้องการลอกจอภาพกราฟิกขนาด 640×348 จุด
A>printscn 480  ถ้าต้องการลอกจอภาพกราฟิกขนาด 640×480 จุด
```

เมื่อข้อความปรากฏบนจอก็ให้กดแป้น S หรือ s

หากท่านเพิ่มการตรวจชนิดของเครื่องพิมพ์ก็จะต้องเพิ่มพารามิเตอร์ในการเรียกไฟล์ใช้งานให้ครบตามที่ได้กำหนดไว้ เช่น

```
A>printscn ลอกจอภาพขนาด 640×348 จุดด้วยเครื่องพิมพ์ 9 หัวเข็ม
A>printscn 24 ลอกจอภาพขนาด 640×380 จุดด้วยเครื่องพิมพ์ 24 หัวเข็ม
A>printscn 480 ลอกจอภาพขนาด 640×480 จุดด้วยเครื่องพิมพ์ 9 หัวเข็ม
A>printscn 480 24 ลอกจอภาพขนาด 640×480 จุดด้วยเครื่องพิมพ์ 24 หัวเข็ม เป็นต้น
```

2. รันไฟล์ที่ให้ภาพกราฟิกบนจอภาพที่เราต้องการลอกจอภาพ

3. เมื่อภาพที่ต้องการลอกปรากฏบนจอภาพให้ทำดังนี้

กดแป้น Alt+Left Shift เพื่อพิมพ์กราฟิกหน้า 0 หากปรากฏว่าการพิมพ์มีแต่กระดาษว่างไม่ได้ภาพตามที่ปรากฏบนจอภาพ ก็แสดงว่าภาพนั้นอยู่ในกราฟิกหน้า 1 ท่านต้องกดแป้น Alt+Right Shift จึงได้ภาพตามต้องการ

4. ถ้าต้องการยกเลิกการเป็นโปรแกรมเรสซิเดนต์ก็ให้กดแป้น Alt+Shift โดยการยกเลิกตามข้อ 4 นี้ ไม่ได้ยกเลิกหน่วยความจำที่โปรแกรมนี้ฝังตัวอยู่ แต่เป็นการยกเลิกการทำงานของโปรแกรมเท่านั้น

ผู้เขียนลองใช้โปรแกรมนี้พิมพ์ภาพจากโปรแกรมที่ใช้เครื่องอ่านภาพ (scanner) เข้ามา ซึ่งโปรแกรมพวกนี้จะมีขนาดใหญ่และใช้กราฟิกขนาด 640×480 จุด โดยได้ทำการอ่านภาพของดาราไทยคนหนึ่งเข้ามาให้แสดงบนจอภาพ พร้อมกับสั่งให้กลับสีของภาพแล้วพิมพ์ลงกระดาษด้วยเครื่องพิมพ์ 24 หัวเข็ม ซึ่งจะได้ภาพบนกระดาษดังรูปที่ 3 โดยเป็นการลอกที่กราฟิกหน้า 0 คือกดแป้น Alt+Left Shift พร้อมกัน

ส่วนรูปที่ 4 เป็นการพิมพ์ภาพหน้าจอจากโปรแกรมช่วยสอนภาษาอังกฤษสำหรับเด็ก ซึ่งเป็นโปรแกรมที่เขียนบนกราฟิกหน้า 0 ขนาด 640×348 จุด ซึ่งจะเห็นได้จากความสูงของภาพจะไม่เท่ากับภาพที่ 6

หวังว่า...การพิมพ์ภาพกราฟิกครั้งต่อไป คงจะไม่ใช่เรื่องยุ่งยากอีกต่อไป พบกันใหม่ในตอนหน้าด้วยเรื่องการเก็บภาพหน้าจอลงแผ่นบันทึก